

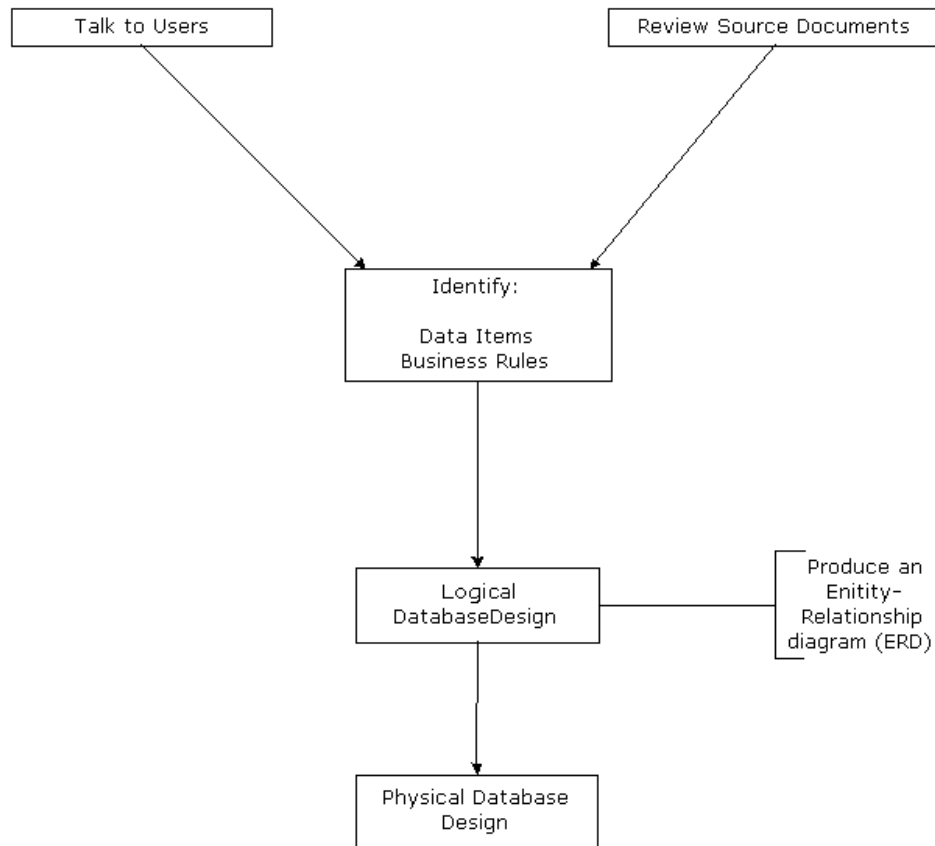
SDEV1201

LESSON 20

NORMALIZATION

ERD Review

The Database Design Process



The database design process is a methodology used to design the structure of a database for an application. Prior to starting the database design process the designer must understand:

- The purpose of the database (i.e. Why does one need the database?).
- The data that will be stored in the database.
- What functions the database will support (e.g. add a customer record).

The database designer develops this understanding by completing systems analysis and design tasks such as:

- Talking to the users to identify requirements.
- Reviewing source documents (e.g. reports, screen layouts, paper-based business forms).

Create a logical database design

CustomerNumber	LastName	FirstName	Phone
1	Jones	Abby	439 - 1763
2	Davis	James	485 - 4309

The next step is to create a logical database design. A logical database design represents the programmer's and the user's perception of the corporate data stored in the database. For example, when using a relational database management system, data is organized into a collection of tables (rows and columns) regardless of the actual arrangement of the data on the physical storage device (e.g. a hard-drive). A Customer table is used to store data about customers as depicted above.

Each row in the table represents one specific customer. Each column in the table represents one attribute or characteristic of a customer.

Entity Relationship Model

The **Entity-Relationship** model is a tool that organizes and documents the logical design of a database. The entity-relationship model describes the data used by an application in terms of: **entities, attributes, and relationships.**

Entity Relationship Model

Entity

An entity is a person, place, thing or concept that is used in the business context of the application and for which data is collected.

Attributes

A piece of data that describes an entity is called an **attribute**. Synonyms for attribute include: element, property and field. In the order example, the customer entity has the following attributes:

- Customer Number
- Last Name
- First Name
- Address
- Phone Number

The values for all attributes describing an entity make up one instance of the entity. For example, the following values describe one specific customer.

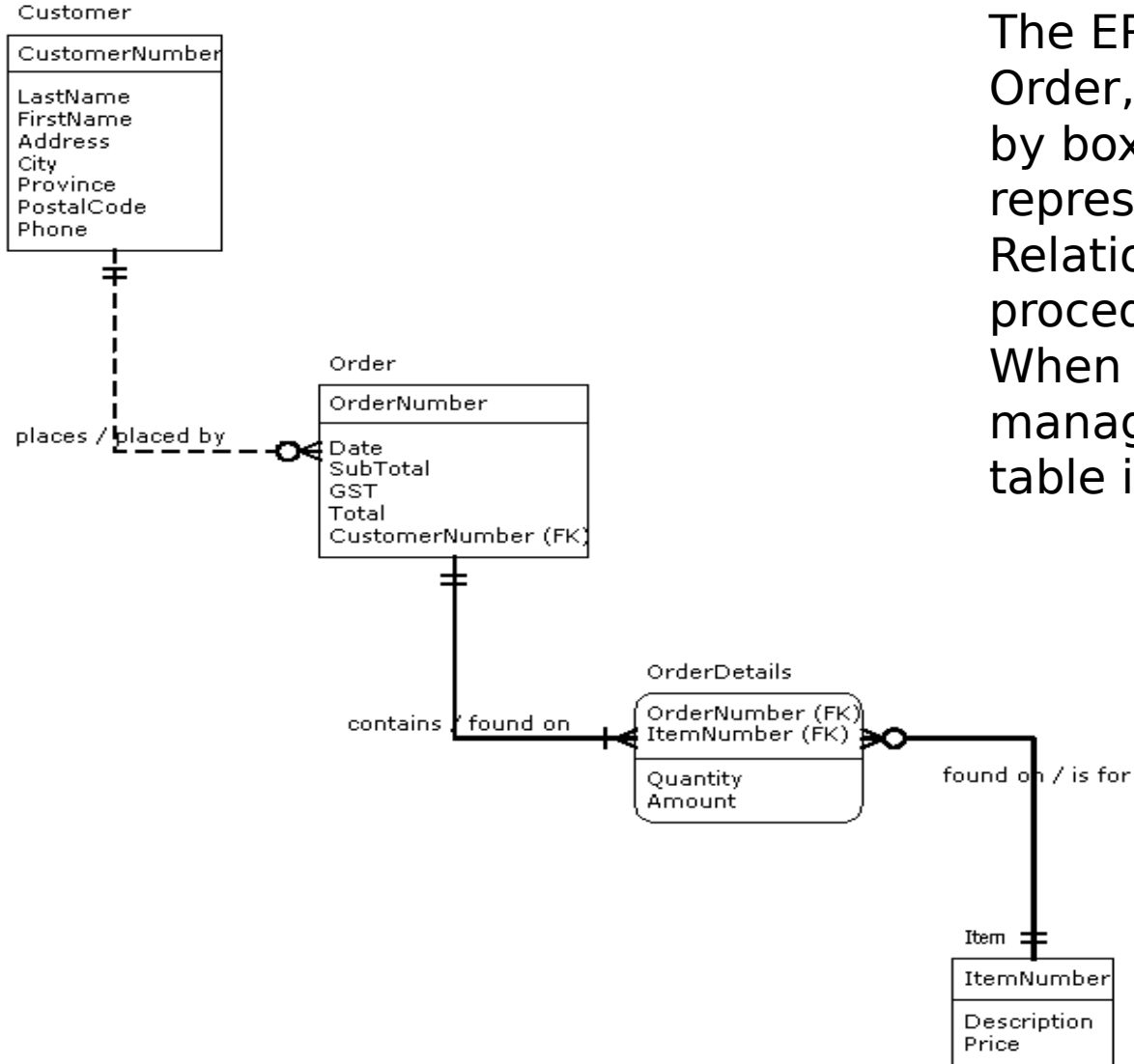
Customer Number	Last Name	First Name	Address	Phone
100	Adams	Joan	9825-77 Ave Edm Ab T5C 8E2	488-8712

Entity Relationship Model

The logical database design process uses the entity-relationship model to represent the business scenario as a collection of entities and relationships. The entity-relationship diagram (ERD) documents the logical design and becomes a starting point for the physical design process, which defines the structure of the database.

The **Entity-Relationship** model is a tool that organizes and documents the logical design of a database. The entity-relationship model describes the data used by an application in terms of: **entities, attributes, and relationships.**

Sample ERD



The ERD shows four entities: Customer, Order, OrderDetails and Item, all represented by boxes. The lines connecting the entities represent relationships between the entities. Relationships reflect the business rules and procedures in place within an organization. When using a relational database management system each entity becomes a table in the database.

Primary Keys

Primary keys are a **unique** identifier for each instance of an entity. e.g. the Item Number uniquely identifies each item in inventory, and can act as the Primary Key for the Item entity.

Item

ItemNumber	Description	Price
333	Hammer	12.00
777	Saw	15
888	Power Drill	90

A combination of two or more attributes acting as the primary key is called a **concatenated** primary key.

OrderDetails

OrderNumber	ItemNumber	Quantity	Price	Amount
1	333	3	15	45
2	777	7	10	70
3	888	8	5	40

Relationships

A relationship represents a business association between two entities. Relationships are based on the business rules of the organization. All relationships are bi-directional in that they can be interpreted from the point of view of each entity. In the customer order example a relationship exists between the customer entity and the order entity. This reflects the fact that each time a customer makes a purchase an order is created. From the customer perspective, one customer can place many orders. From the order perspective, one and only one customer places an order.

Types of Relationships

- One-to-one

- These are very rare.
- e.g. each country has *one UN representative*, and each UN representative represents only *one country*. The Country entity and the UNRepresentative entity have a 1:1 relationship.

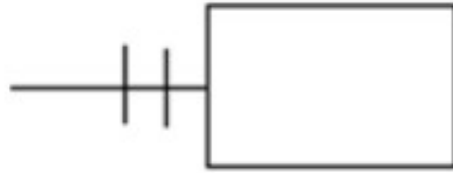
- One-to-many

- We'll see these LOTS.
- e.g. each Department has *many employees*, but each employee belongs to a *single department*. The Employee entity and the Department entity have a 1:many relationship.

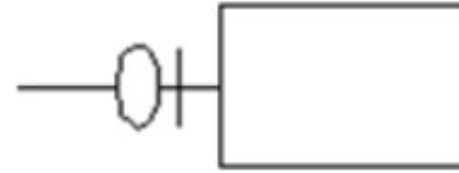
- Many-to-many

- We'll see these LOTS, and we'll do something special in our design to represent them.
- e.g. a student enrolls in *many classes*, and each class has *many students*. The Student entity and the Class entity have a many:many relationship.

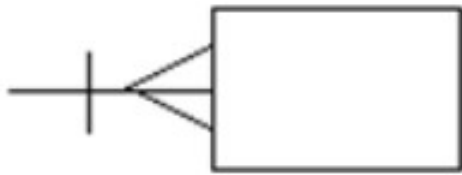
Symbols for Cardinality



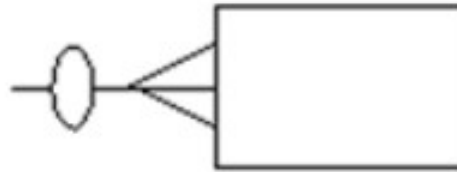
... to one
(mandatory 1)



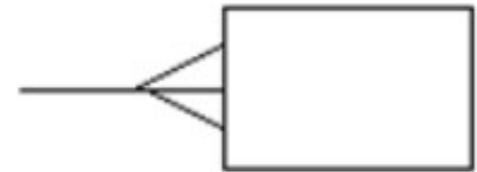
... to zero or one
(optional 1)



... to one or many
(mandatory many)



... to zero, one, or many
(optional many)



... to many

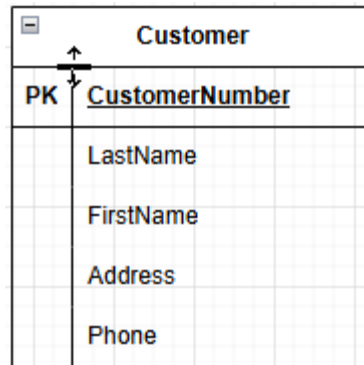
Foreign keys

A **foreign key** is how we define relationships between entities.

A foreign key is a **primary key of one entity** (the “parent”) that appears as an **attribute of another entity** (the “child”).

Note: the two attributes may or may not have the same name. e.g. the Student ID in the Marks table may be called just ID in the Student table.

Entities and Tables



After the ERD design is complete, when the database is created the entities become tables.

<u>CustomerNumber</u>	FirstName	<u>LastName</u>	Address	Phone
100	Adams	Julie	9825-77 Ave Edmonton AB T5C 8E2	780-111-1111
200	Smith	Miles	4308-133 Ave Edmonton AB T5C 8E9	780-222-2222

Today's Objective

By the end of this section you will be able to understand:

- ☐ Why normalization is necessary.
- ☐ Normalization procedure.

Important: Quiz 4 on Stored Procedures and Triggers will be next **Monday, November 24, 2025.**

Normalization: What & Why?

FOR REFERENCE SEE THE “01 NORMALIZATION WHAT AND WHY” WORD DOCUMENT LOCATED IN THE “WEEK 13 – NORMALIZATION” SECTION OF BRIGHTSPACE UNDER MATERIALS.

Normalization: what & why?

When designing a database, two critical question are:

- **How many** tables do we need?
- **What data** belongs in each table?

Initially, db designers did this intuitively. Group “like” with “like”. All info about employees goes in the **Employee** table.

However, what’s intuitive to one might not be intuitive to another.

The problem was there was no systematic method to help database designers decide how many tables there should be and what data items should be in which tables.

What is normalization?

The rules of normalization as designed by Edgar F. Codd, guide us to decide:

- The **number** of tables
- The **assignment of data items** (attributes) to the various tables

Normalization is the industry standard.

Evaluating database design

- Does the design minimize **redundant data**?
- Does the design **minimize the occurrence of anomalies** within the data?

3 major types of anomalies: **update**, **insert**, and **delete**.

Minimizing Redundant Data

SSN	Name	BDate	DeptNum	DeptName	DeptMgrSSN
123456789	Smith, John B	09-Jan-55	5	Research	334455667
234567890	Wong, Franklin	08-Dec-45	5	Research	334455667
345678901	Zelaya, Alicia	19-Jul-58	4	Admin	223344556
456789012	Narayan, Remesh	15-Sep-62	4	Admin	223344556
567890123	English, Joyce	31-Jul-53	5	Research	334455667
678901234	Borg, James	10-Nov-27	5	Research	334455667
789012345	Jabbar, Ahmand	29-Mar-59	4	Admin	223344556
890123456	Wallace, Jennifer	20-Jun-64	1	Marketing	445566778

Applying the rule of normalization:

SSN	Name	BDate	DeptNum
123456789	Smith, John B	09-Jan-55	5
234567890	Wong, Franklin	08-Dec-45	5
345678901	Zelaya, Alicia	19-Jul-58	4
456789012	Narayan, Remesh	15-Sep-62	4
567890123	English, Joyce	31-Jul-53	5
678901234	Borg, James	10-Nov-27	5
789012345	Jabbar, Ahmand	29-Mar-59	4
890123456	Wallace, Jennifer	20-Jun-64	1

DeptNum	DeptName	DeptMgrSSN
5	Research	334455667
4	Admin	223344556
1	Marketing	445566778

Minimizing Update Anomalies

SSN	Name	BDate	DeptNum	DeptName	DeptMgrSSN
123456789	Smith, John B	09-Jan-55	5	Research	222222222
234567890	Wong, Franklin	08-Dec-45	5	Research	222222222
345678901	Zelaya, Alicia	19-Jul-58	4	Admin	223344556
456789012	Narayan, Remesh	15-Sep-62	4	Admin	223344556
567890123	English, Joyce	31-Jul-53	5	Research	222222222
678901234	Borg, James	10-Nov-27	5	Research	334455667
789012345	Jabbar, Ahmand	29-Mar-59	4	Admin	223344556
890123456	Wallace, Jennifer	20-Jun-64	1	Marketing	445566778

Minimizing Update Anomalies

SSN	Name	BDate	DeptNum
123456789	Smith, John B	09-Jan-55	5
234567890	Wong, Franklin	08-Dec-45	5
345678901	Zelaya, Alicia	19-Jul-58	4
456789012	Narayan, Remesh	15-Sep-62	4
567890123	English, Joyce	31-Jul-53	5
678901234	Borg, James	10-Nov-27	5
789012345	Jabbar, Ahmand	29-Mar-59	4
890123456	Wallace, Jennifer	20-Jun-64	1

DeptNum	DeptName	DeptMgrSSN
5	Research	222222222
4	Admin	223344556
1	Marketing	445566778

Minimizing Insert Anomalies

SSN	Name	BDate	DeptNum	DeptName	DeptMgrSSN
111111111	Doe, John	17-May-70	5	Admin	334455667
123456789	Smith, John B	09-Jan-55	5	Research	334455667
234567890	Wong, Franklin	08-Dec-45	5	Research	334455667
345678901	Zelaya, Alicia	19-Jul-58	4	Admin	223344556
456789012	Narayan, Remesh	15-Sep-62	4	Admin	223344556
567890123	English, Joyce	31-Jul-53	5	Research	334455667
678901234	Borg, James	10-Nov-27	5	Research	334455667
789012345	Jabbar, Ahmand	29-Mar-59	4	Admin	223344556
890123456	Wallace, Jennifer	20-Jun-64	1	Marketing	445566778

Minimizing Insert Anomalies

SSN	Name	BDate	DeptNum
111111111	Doe, John	17-May-70	5
123456789	Smith, John B	09-Jan-55	5
234567890	Wong, Franklin	08-Dec-45	5
345678901	Zelaya, Alicia	19-Jul-58	4
456789012	Narayan, Remesh	15-Sep-62	4
567890123	English, Joyce	31-Jul-53	5
678901234	Borg, James	10-Nov-27	5
789012345	Jabbar, Ahmand	29-Mar-59	4
890123456	Wallace, Jennifer	20-Jun-64	1

DeptNum	DeptName	DeptMgrSSN
5	Research	334455667
4	Admin	223344556
1	Marketing	445566778

Data dilemma with a non-normalized database

Suppose the company expands and creates a new department. Under the intuitive design the new department information cannot be recorded in the database until the first employee is hired for the department (the employee social security number is the primary key for the EmpDept table). With the normalized design new departments can be added any time because the department information is kept in its own table.

Minimizing Delete Anomalies

A delete anomaly occurs when we delete data from the database causing an inconsistency in the data.

Suppose all employees in the research department leave and we need to hire new employees to replace them in the near future. Under the intuitive design when the last employee for the research department is deleted we lose the information about the research department itself. Deleting employee information should not cause a loss of information about a department.

With the normalized design we are free to delete employee information without losing department information because each is stored in its own table.

Break

Normalization

FOR REFERENCE SEE THE “02 NORMALIZATION HOW WORD” WORD DOCUMENT LOCATED IN THE “WEEK 13 - NORMALIZATION” SECTION OF BRIGHTSPACE UNDER MATERIALS.

Normalization: How

The normalization process provides database designers with a systematic method of breaking down large tables into a group of **related, smaller** tables which:

- Are **easier** to understand
- **Minimize the amount of redundant data** stored in the database
- **Minimize the likelihood of anomalies** occurring within the data.

Normalization states

- First normal form (1NF)
- Second normal form (2NF)
- Third normal form (3NF)
- Boyce-Codd normal form
- Fourth normal form (4NF)
- Domain-Key normal form

Each normal form represents a set of rules.

Before we start...

A note on business rules

The normalization process

1 Collect data

1. Collect **views** (subsets of data) using source documents and business rules.

- Before applying the rules, you should have a clear idea of the entities, relationships, and candidate keys.

2 Normalize

2. Using a view, apply the rules of 1NF, 2NF, and 3NF to produce a **schema**.

- Repeat step #2 for each view.

3 Merge

3. **Merge** all those schemas into a single schema.

- Tables with the **same** PK can be merged together. All entities, attributes, and relationships need to be preserved.

Normalization Process

1NF (First Normal Form) has two rules:

1. A table MUST contain atomic attributes.
2. A table cannot contain any repeating groups of attributes.

2NF has one rule:

1. All non-key attributes in a table must fully depend on the entire primary key of the table.

3NF has one rule:

1. A non-key attribute cannot be fully dependent on another non-key attribute.

Normalization Reference Sheet

In the “Week 13 – Normalization” section of Brightspace in Materials there is the Normalization Reference Sheet. It has the terms in more basic understandable format. It also has all the steps of the normalization procedure which if followed, works well.

Let's practice!

Employee Workload

Employee Id: 1111111 Name: John Doe

Works For Department Number: 1 Name: Information Systems

Currently Assigned to:

Project		Department	Weekly Hours
Number	Name		
IS-101	2000 Bug	Information Systems	20
PR-222	Inventory Control	Warehousing	5

Business Rules

1. An employee works for a single department.
2. Departments have many employees.
3. A single department develops a project.
4. A department can develop many projects.
5. An employee can be assigned to a maximum of 4 projects.
6. Projects can have many employees.
7. Employee ID, Department number, and project number uniquely identify employees, departments, and projects respectively.

Normalization Example

Before applying the rules of normalization you should have a clear understanding of the entities, relationships and candidate keys in the view.

Entities: Employee, Department, Project

Relationships: 1. Employee : Department

M : 1

2. Department : Project

1 : M

3. Employee : Project

M : M

Candidate Keys: Employee Id, Department Number, Project Number

We can now apply the normalization process to produce the schema (database design) for the view.

Step 0: Create the initial table

Identify the PK and any repeating group(s) of attributes.

- PKs must **always** be unique, and may be made up of more than 1 column.
- A repeating group is 1+ attributes that can have multiple values for a single instance of an entity *at the same time*. The repeating group should be in parenthesis ().

Step 0: Create the initial table *

The employee workload view has three candidate keys. We must select a primary key from the candidate keys. The view contains information about a specific employee; this points to the Employee Id as the primary key. We should consider all candidates though. The department number will not uniquely identify a row because a single department can have many employees assigned to it. The project number will not uniquely identify a row because many employees can work on a project. Each row in the table records information about a specific employee, the employee id will uniquely identify a row, so employee id **MUST** act as the primary key for the initial table.

Step 0: Create the initial table

ONF:

Employee Table

EmployeeID (PK), Name, DepartmentNumber,
DepartmentName, (ProjectNumber, ProjectName,
SponsoringDepartmentName, WeeklyHours)

Step 1: Apply the rules of 1NF *

1. A table MUST contain only **atomic** attributes.
2. A table **cannot contain any repeating groups of attributes**.

If a **repeating group** of attributes exists:

3. Move the repeating group of attributes to a **new table**.
4. Copy the primary key of original and place in the new table as a foreign key.
5. Designate cardinality* of relationship between entities.
6. Designate a primary key for the new table.

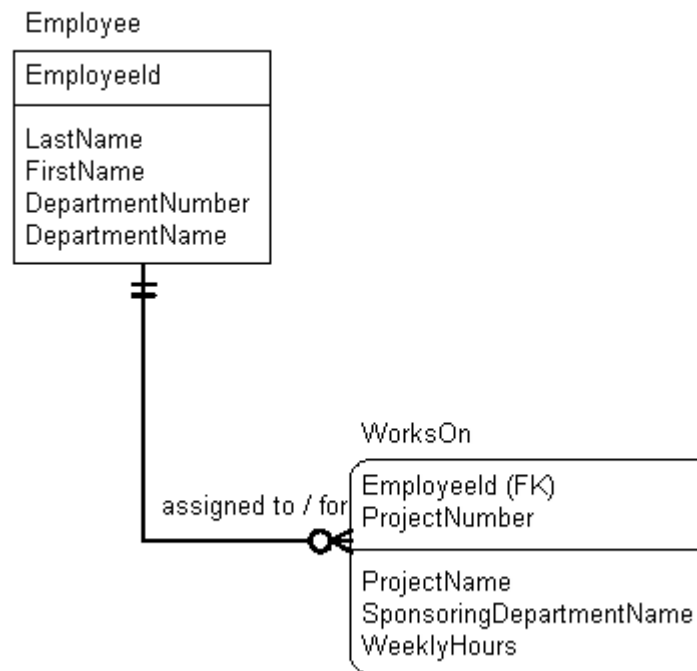
Step 1: Apply the rules of 1NF

1NF:

EmployeeID (PK), **FirstName**, **LastName**,
DepartmentNumber, DepartmentName

EmployeeID (PK)(FK), **ProjectNumber (PK)**,
ProjectName, SponsoringDepartmentName,
WeeklyHours

Step 1: Apply the rules of 1NF



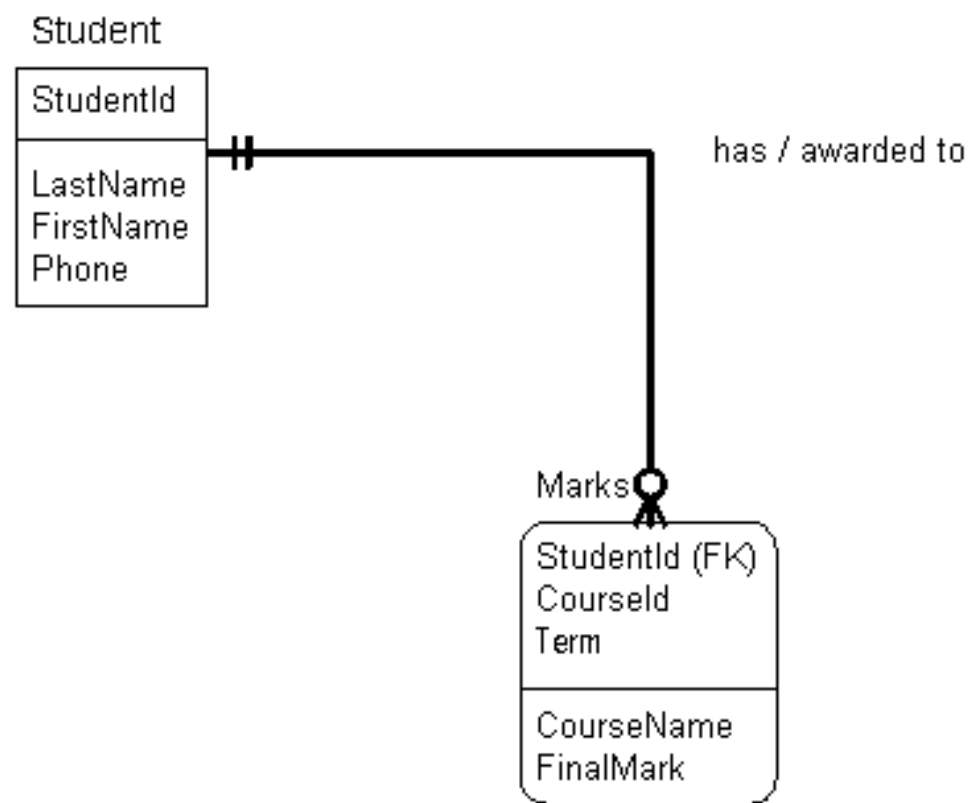
Step 2: Apply the rules of 2NF

1. All non-key attributes in a table must **fully depend on the entire primary key** of the table.

A violation can only occur in a table with a composite primary key: if we don't have one, we're done with 2NF!

Are there any attributes that only rely on part of the PK? If so, we have a violation!

Consider the following tables:

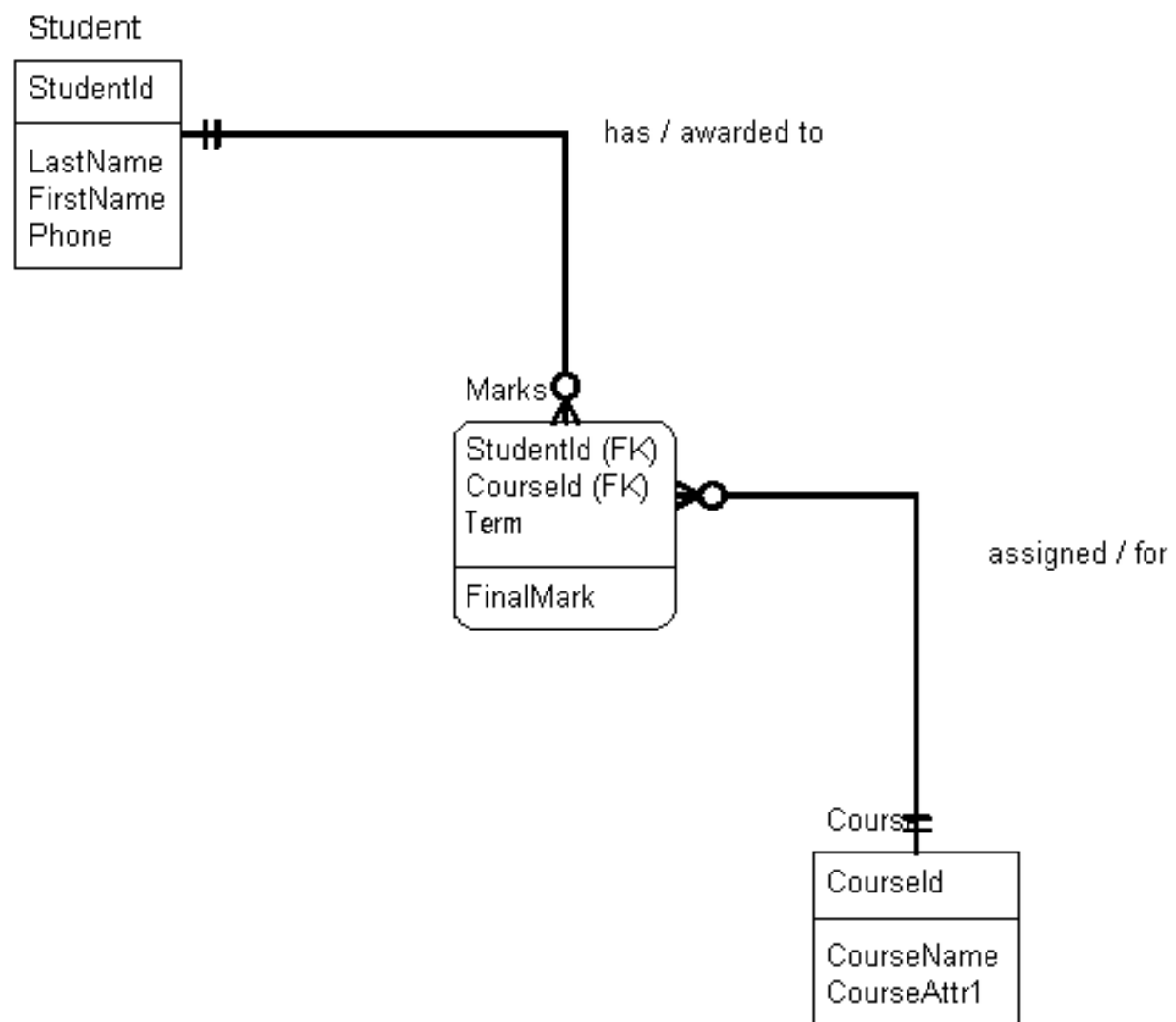


Step 2: Apply the rules of 2NF

An attribute that violates 2NF is referred to as a partial dependency.

If a partial dependency exists:

1. Move the partially dependent attribute to a new table.
2. Duplicate the part of the primary key it's dependent on, and place in new table as the primary key (it's now a foreign key in the original table).
3. Designate cardinality of the relationship.



Now apply the 2NF rule to
Employee Workload. *

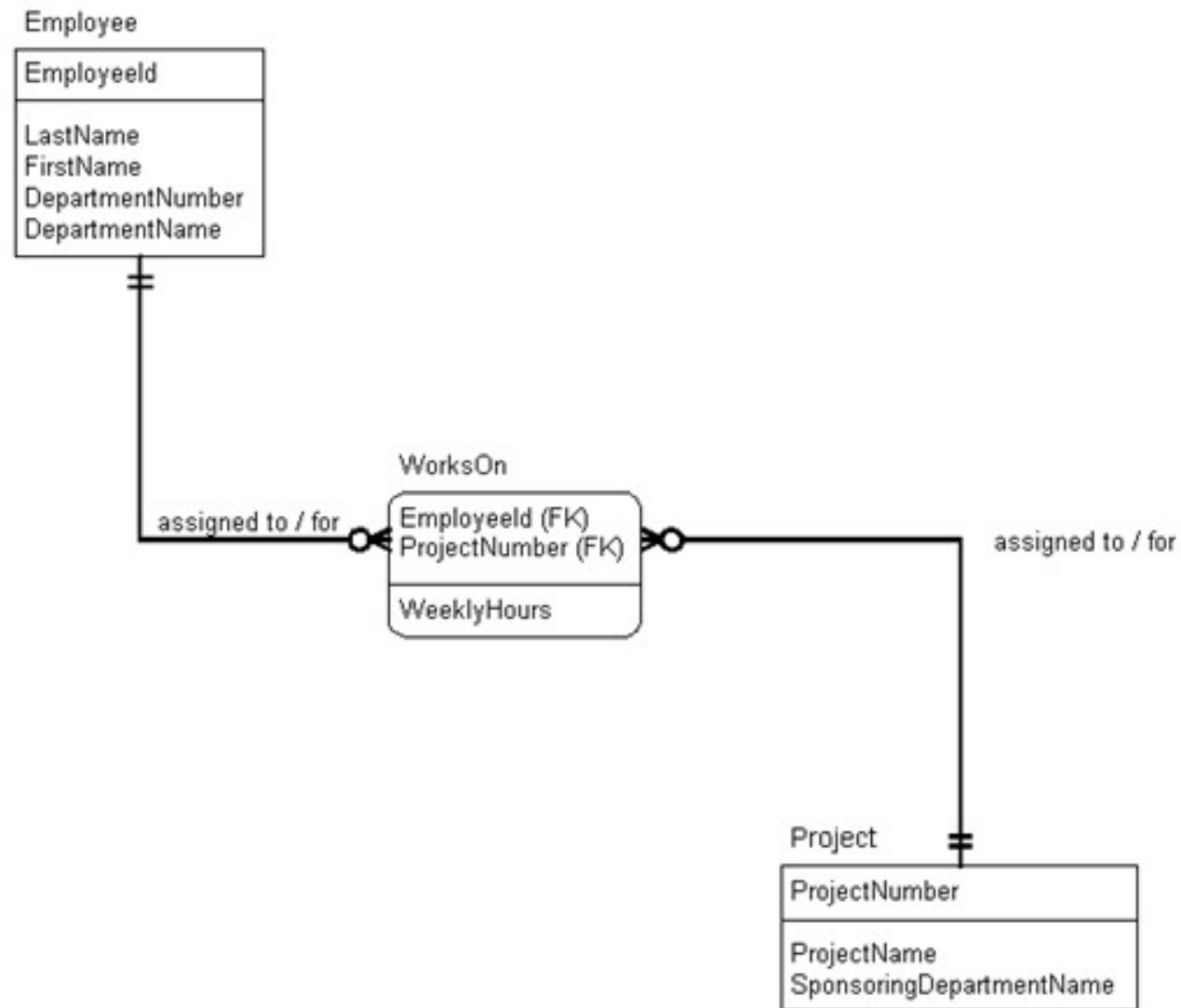
Step 2: Apply the rules of 2NF

2NF:

EmployeeID (PK), FirstName, LastName, DepartmentNumber,
DepartmentName

EmployeeID(PK)(FK), ProjectNumber (PK)(FK), WeeklyHours

ProjectNumber(PK), ProjectName, SponsoringDepartment



Step 3: Apply the rules of 3NF

1. A non-key attribute **cannot be fully dependent** on another non-key attribute.

A non-key attribute that is fully dependent on another non-key attribute (instead of the PK) is termed a **transitive dependency**. To fix:

2. Move the transitive dependent attribute to a **new table**.
3. Duplicate the non-key attribute it was dependent on and place in new table as the primary key (it's now the foreign key in the original table).
4. Designate cardinality of the relationship.

Now apply the 3NF rule to
Employee Workload. *

Step 3: Apply the rules of 3NF

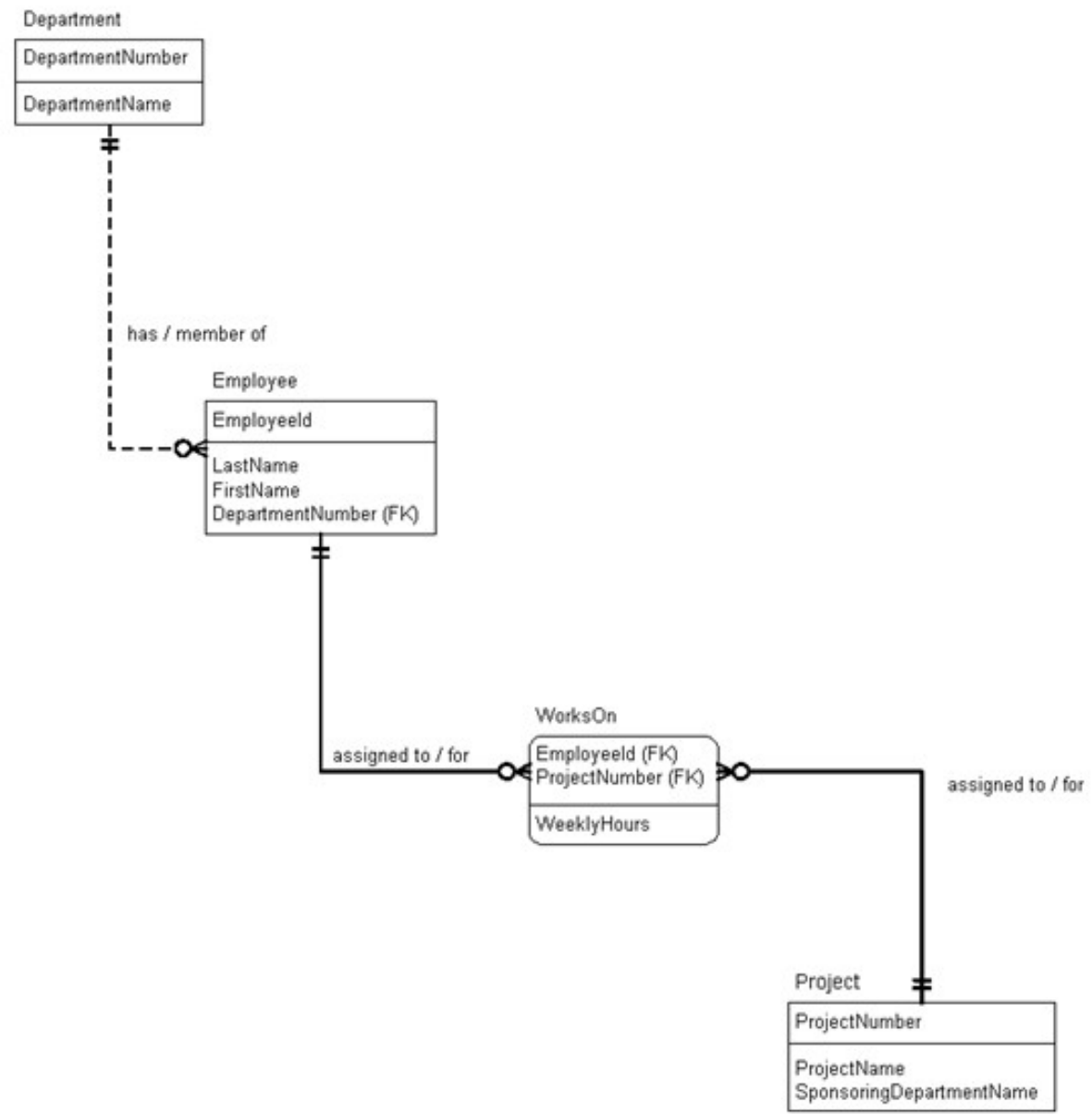
3NF:

EmployeeID (PK), FirstName, LastName, *DepartmentNumber (FK)*

DepartmentNumber (PK), DepartmentName

EmployeeID (PK)(FK), **ProjectNumber (PK)(FK)**,
WeeklyHours

ProjectNumber (PK), ProjectName,



Normalization

☐ 1NF

- ☐ A table must contain atomic attributes.
- ☐ A table cannot contain any repeating groups of attributes.

☐ 2NF

- ☐ All non-key attributes in a table must fully depend on the entire primary key of the table.

☐ 3NF

- ☐ A non-key attribute cannot be fully dependent on another non-key attribute.

Normalization Steps – Initial Table

For each view:

1. Create the initial table:

- a. List all the attributes in the view
- b. Designate the Primary Key
- c. Show any repeating groups in parenthesis (brackets)

Normalization Steps – 1NF

2. First Normal Form (all atomic attributes and no repeating groups)

- a. If composite attributes exist replace them with atomic attributes
- b. If repeating groups exist:
 - i. Remove the repeating groups and place them in a new table
 - ii. Duplicate the Primary Key from the original table and place it in the new table as a Foreign Key joining the new table to the original table

iii. Designate a Primary Key for the new table

Normalization Steps – 2NF

3. Second Normal Form (No Partial Dependencies)

a. If Partial Dependencies exists:

- i. Remove the attribute(s) that can be uniquely identified by just part of the Primary Key and place them in a new table
- ii. Duplicate the part of the Primary Key that could uniquely identify those attribute(s) and place it in the new table as the Primary Key
- iii. The part of the Primary Key that uniquely identified the attributes (the Primary Key in the new table) becomes a Foreign Key in the original table.

Normalization Steps – 3NF

4. Third Normal Form (No Transitive Dependencies)

a. If Transitive Dependencies exist:

- i. Remove the attribute(s) that could be uniquely identified by another non key attribute and place them in a new table
- ii. Duplicate the non key field that could uniquely identify those attribute(s) and place it in the new table as the Primary Key
- iii. The non key field that uniquely identified the attributes (the primary key in the new table) becomes a foreign key in the original table

Recommended

From the Brightspace site in “Week 13 – Normalization” in Materials access and review the following:

- 01 Normalization What and Why
- 02 Normalization How

Important: Quiz 4 on Stored Procedures and Triggers will be next **Monday, November 24, 2025.**